```
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
FractionToCF <- function(numerator, denominator){
  num <- numerator; den <- denominator
  #
  i <- 0
  cf <- vector()
  if(num == 1) cf <- c(0, den)
  while(num > 1){
    i <- i+1
    jj <- floor(num/den)
    cf[i] <- jj
    temp <- num
    num <- den
    den <- temp-floor(temp/den)*den}
  return(cf)
}
#
# # # # # # # # # # # # # # # # # # # # # # #
#
#   system = 1  Stern-Brocot
#   system = 2  Bird
#   system = 3  HCS
#   system = 4  Yurramendi-2
#
CFToBinary.2.1 <- function(cf){
  binary.vector <- 1
  for(k in 1:length(cf))
    binary.vector <- c(binary.vector, rep(k%%2, cf[k]))
  binary.vector <- binary.vector[-sum(cf)]
  return(binary.vector)
}
#
CFToBinary.2.2 <- function(cf){
  binary.vector <- 1
  for(k in 1:length(cf))
    binary.vector <- c(binary.vector, rep(k%%2, cf[k]))
  binary.vector <- binary.vector[-sum(cf)]
  filter   <- rep(c(1,0), 15)[1:sum(cf)]
  binary.vector <- as.numeric(binary.vector != filter)
  binary.vector[1] <- 1
  return(binary.vector)
}
#
CFToBinary.2.3 <- function(cf){
  binary.vector <- rep(0, sum(cf))
  binary.vector[cumsum(cf)] <- 1
  ifelse(cf[1] == 0,
         binary.vector <- c(1, 0, binary.vector[-(sum(cf)-0:1)]),
         binary.vector <- c(1, 1, binary.vector[-(sum(cf)-0:1)]))
  return(binary.vector)
}
#
CFToBinary.2.4 <- function(cf){
  binary.vector <- rep(1, sum(cf))
  binary.vector[cumsum(cf[-length(cf)]) + 2] <- 0
  return(binary.vector)
}
# # # # # # # # # # # # # # # # # # # # # # #
#
BinaryToLocation <- function(binary.vector){
  location <- strtoi(paste(binary.vector, collapse = ""), base = 2)
  return(location)
}
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
FractionToLocation.CF.2 <- function(system, numerator, denominator){
#
#   system = 1  Stern-Brocot
#   system = 2  Bird
#   system = 3  HCS
#   system = 4  Yurramendi-2
#
# Fraction --> Continued Fraction --> Binary --> Location
#
# Fraction --> Continued Fraction
#
  cf <- FractionToCF(numerator, denominator)
#
#              Continued Fraction --> Binary
#
  if(system == 1)  binary.vector <- CFToBinary.2.1(cf)
  if(system == 2)  binary.vector <- CFToBinary.2.2(cf)
  if(system == 3)  binary.vector <- CFToBinary.2.3(cf)
  if(system == 4)  binary.vector <- CFToBinary.2.4(cf)
#
#                                  Binary --> Location
#
  location <- BinaryToLocation(binary.vector)
#
  return(location)
}
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
# Examples: Compute location of 9/5 in all four systems
#
FractionToLocation.CF.2(1, 9, 5) # 55
FractionToLocation.CF.2(2, 9, 5) # 61
FractionToLocation.CF.2(3, 9, 5) # 60
FractionToLocation.CF.2(4, 9, 5) # 51
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
```